Claims

What is claimed is:

1.   A method of evaluating a software component, comprising:

      providing an abstraction matrix that describes the software component, the abstraction matrix comprising at least one test case scenario and mapped expected results therefore;

      testing the software component using test cases to generate test results; and

      evaluating the test results using the abstraction matrix, the evaluating including comparing a test case employed in the testing to the at least one test case scenario of the abstraction matrix and if a match is found, comparing the test result for that test case with the mapped expected result therefore in the abstraction matrix.

2.   The method of claim 1, where the evaluating comprises automatically evaluating the test results using the abstraction matrix.

3.   The method of claim 1, further comprising, prior to said testing, deriving at least some test cases from the at least one test case scenario of the abstraction matrix.

4.   The method of claim 1, wherein the software component comprises multiple states, and wherein the abstraction matrix comprises multiple test case scenarios, each test case scenario being associated with a different state of the software component.

5.   The method of claim 1, further comprising generating an error log, said error log containing a list of test cases each having a test result which failed to match the mapped expected result therefore in a matching test case scenario of the abstraction matrix.

6.   The method of claim 5, wherein said error log further includes the test results of the failing test cases, as well as the mapped expected results from the abstraction matrix for the failing test cases.

7.   The method of claim 1, further comprising creating at least one test results file and at least one abstraction matrix file, and wherein said evaluating comprises automatically reviewing each test result in the at least one test results file by comparing its test case to the at least one test case scenario of the abstraction matrix contained in the at least one abstraction matrix file.

8.   The method of claim 1, wherein the providing comprises creating the abstraction matrix from a functional specification of the software component.

9.   The method of claim 1, wherein the evaluating comprises automatically extracting test statistics, said test statistics including a total number of test cases executed, and at least one of a total number of test cases successfully executed or a total number of test cases unsuccessfully executed.

10.   The method of claim 1, wherein the testing of the software component is based on layers of the software component, and wherein the evaluating comprises evaluating the test results for at least one layer of the software component using the abstraction matrix.

11.   The method of claim 10, wherein the providing comprises providing the abstraction matrix to include at least one test case scenario for each layer of the software component.

12.   A system for evaluating a software component
comprising:

an abstraction matrix that describes the software
component, the abstraction matrix comprising at least
one test case scenario and mapped expected results
therefore;

means for testing the software component using
test cases to generate test results; and

means for evaluating the test results using the
abstraction matrix, the means for evaluating including
means for comparing a test case to the at least one
test case scenario of the abstraction matrix and if a
match is found, for comparing the test result for that
test case with the mapped expected result therefore in
the abstraction matrix.

13.   The system of claim 12, where the means for
evaluating comprises means for automatically evaluating the
test results using the abstraction matrix.

14.   The system of claim 12, further comprising means
for deriving at least some test cases from the at least one
test case scenario of the abstraction matrix for use by the
means for testing.

15.   The system of claim 12, wherein the software
component comprises multiple states, and wherein the
abstraction matrix comprises multiple test case scenarios,
each test case scenario being associated with a different
state of the software component.

16.   The system of claim 12, further comprising means
for generating an error log, said error log containing a
list of test cases each having a test result which failed to
match the mapped expected result therefore in a matching
test case scenario of the abstraction matrix.

17.   The system of claim 16, wherein said error log
further includes test results of the failing test cases, as
well as mapped expected results from the abstraction matrix
for the failing test cases.

18. The system of claim 12, further comprising means for creating at least one test results file and at least one abstraction matrix file, and wherein said means for evaluating comprises means for automatically reviewing each test result in the at least one test results file by comparing its test case to the at least one test case scenario of the abstraction matrix contained in the at least one abstraction matrix file.

19. The system of claim 12, wherein the means for providing comprises means for creating the abstraction matrix from a functional specification of the software component.

20. The system of claim 12, wherein the means for evaluating comprises means for automatically extracting test statistics, said test statistics including a total number of test cases executed, and at least one of a total number of test cases successfully executed or a total number of test cases unsuccessfully executed.

21. The system of claim 12, wherein the means for testing comprises means for testing the software component based on layers of the software component, and wherein the means for evaluating comprises means for evaluating the test results for at least one layer of the software component using the abstraction matrix.

22.   The system of claim 21, wherein the means for providing comprises means for providing the abstraction matrix to include at least one test case scenario for each layer of the software component.

23. A system for evaluating a software component
comprising:

an abstraction matrix that describes the software
component, the abstraction matrix comprising at least
one test case scenario and mapped expected results
therefore;

a first computing unit adapted to test the
software component using test cases to generate test
results; and

a second computing unit adapted to evaluate the
test results using the abstraction matrix, the
evaluating including comparing a test case to the at
least one test case scenario of the abstraction matrix
and if a match is found, comparing the test result for
that test case with the mapped expected result
therefore in the abstraction matrix.

24. The system of claim 23, wherein the first
computing unit and the second computing unit comprise a
single computing unit.

25.   At least one program storage device readable by a machine, tangibly embodying at least one program of instructions executable by the machine to perform a method of evaluating a software component, comprising:

providing an abstraction matrix that describes the software component, the abstraction matrix comprising at least one test case scenario and mapped expected results therefore;

testing the software component using test cases to generate test results; and

evaluating the test results using the abstraction matrix, the evaluating including comparing a test case employed in the testing to the at least one test case scenario of the abstraction matrix and if a match is found, comparing the test result for that test case with the mapped expected result therefore in the abstraction matrix.

26.   The at least one program storage device of claim 25, where the evaluating comprises automatically evaluating the test results using the abstraction matrix.

27.   The at least one program storage device of claim 25, further comprising, prior to said testing, deriving at least some test cases from the at least one test case scenario of the abstraction matrix.

28. The at least one program storage device of claim 25, wherein the software component comprises multiple states, and wherein the abstraction matrix comprises multiple test case scenarios, each test case scenario being associated with a different state of the software component.

29. The at least one program storage device of claim 25, further comprising generating an error log, said error log containing a list of test cases each having a test result which failed to match the mapped expected result therefore in a matching test case scenario of the abstraction matrix.

30. The at least one program storage device of claim 29, wherein said error log further includes test results of the failing test cases, as well as mapped expected results from the abstraction matrix for the failing test cases.

31. The at least one program storage device of claim 25, further comprising creating at least one test results file and at least one abstraction matrix file, and wherein said evaluating comprises automatically reviewing each test result in the at least one test results file by comparing its test case to the at least one test case scenario of the abstraction matrix contained in the at least one abstraction matrix file.

32.   The at least one program storage device of claim
25, wherein the providing comprises creating the abstraction
matrix from a functional specification of the software
component.

33.   The at least one program storage device of claim
25, wherein the evaluating comprises automatically
extracting test statistics, said test statistics including a
total number of test cases executed, and at least one of a
total number of test cases successfully executed or a total
number of test cases unsuccessfully executed.

34.   The at least one program storage device of claim
25, wherein the testing of the software component is based
on layers of the software component, and wherein the
evaluating comprises evaluating the test results for at
least one layer of the software component using the
abstraction matrix.

35.   The at least one program storage device of claim
34, wherein the providing comprises providing the
abstraction matrix to include at least one test case
scenario for each layer of the software component.

*   *   *   *   *